Our File No: 42390P10857
Express Mail No: EM014067152US

UNITED STATES PATENT APPLICATION

FOR

NAK THROTTLING FOR USB HOST CONTROLLERS

Inventors:

John S. Howard
John I. Garney

Prepared By:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN, LLP
12400 Wilshire Blvd., 7th Floor
Los Angeles, California 90025-1026
(310) 207-3800

NAK THROTTLING FOR USB HOST CONTROLLERS

FIELD OF THE INVENTION

This invention relates generally to buses, and more particularly to managing bus resources.

BACKGROUND

The Universal Serial Bus (USB) is a half duplex single logical wire which permits relatively high speed serial communication between a host system bus and devices on the USB. The USB supports "USB devices" such as keyboards, joy sticks, pointing devices, mice and audio speakers. In order to execute transactions (i.e. data transfers) on the bus, the host controller must service a schedule, which is a set of data structures in shared memory. These data structures are known as elements, with one element corresponding to each endpoint/device coupled to the bus. One particular subset of elements is organized into a circular linked list, where each list element describes the control and buffer information for the host controller to conduct USB transactions to a particular device on the USB. The host controller services the schedule by repeatedly traversing the circular linked list executing bus transactions as appropriate. This means the host controller will read a list element, execute a transaction, and then write back the results to the element. It should be noted that a transaction will be executed only if the list element control bits indicate that it is appropriate (e.g. a buffer is available).

Additionally, the USB has flow control built into the bus protocol. Thus, the host controller will issue a request for data (e.g. a read), and the device will return either a data packet or a flow control handshake (e.g. flow control event) indicating that it does not currently have data available. Depending on the circumstances, this flow control event is known as a Nak or a Nyet, per Enhanced Host Controller Interface Specification for USB, Revision .95, November 10, 2000, available on the Internet at the following website: http://developer.intel.com/technology/usb/ehcispec.htm. These flow control events are very short transactions. As described below, this can cause problems.

During operation, devices can issue flow control events much or even most of the time. There are two negative consequences to this behavior. First, the flow control events are very short, and when most, or all, of the devices represented in the schedule are mostly issuing flow control events, the host controller is able to traverse the circular list very quickly, occupying significant system bandwidth in the process. For example, a host controller following the protocol set forth in USB Specification, Version 1.0, January 19, 1996, was measured utilizing 70% of the Peripheral Component Interconnect bus traversing the schedule but not moving any data. Second, the flow control events occupy bus time which could be used by other devices that have data to move.

DESCRIPTION OF THE DRAWINGS

The invention is illustrated by way of example and not by way of limitation in the figures of the accompanying drawings in which like references indicate similar elements. It should be noted that references to "an" or "one" embodiment in this disclosure are not necessarily to the same embodiment, and such references mean at least one.

FIG. 1 is a flow chart of a method of traversing a schedule with a bus master/host controller.

FIG. 2 is a diagram of a circular linked list of elements, which make up a schedule.

FIG. 3 is a diagram of a system according to the present invention.

DETAILED DESCRIPTION

The present invention overcomes the problems in the existing art described above by providing a method and apparatus by which a bus master/host controller skips, during the traversal of a schedule, schedule elements which have issued a threshold number of flow control events. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without some of these specific details. The following description and the accompanying drawings provide examples for the purposes of illustration. However, these examples should not be construed in a limiting sense as they are merely intended to provide exemplary embodiments of the present invention rather than to provide an exhaustive list of all possible implementations of the present invention. In other instances, well-known structures and devices are shown in block diagram form in order to avoid obscuring the details of the present invention.

In one embodiment, the present invention may be provided as a computer program product which may include a machine-readable medium having stored thereon instructions which may be used to program a computer (or other electronic devices) to perform a process according to an embodiment of the present invention. The machine-readable medium may include, but is not limited to, floppy diskettes, optical disks, CD-ROMs (compact disc-read-only memory), and magneto-optical disks, ROMs (read-only memory), RAMs (random access memory), EPROMs (erasable programmable read-only memory), EEPROMs (electrically erasable programmable read-only memory), magnet or optical cards, flash memory, or other type of media / machine-readable medium suitable for storing electronic instructions. Moreover, an embodiment of the present invention may also be downloaded as a computer program product, wherein the program may be transferred from a remote computer (e.g., a server) to a requesting computer (e.g., a client) by way of data signals embodied in a carrier wave or other propagation medium via a communication link (e.g., a modem or network connection).

Referring now to **Figure 1**, a flow chart is shown which illustrates the manner in which an embodiment of the present invention services a schedule (after the first complete traversal of the schedule, discussed below). The bus master or host controller (terms can be used interchangeably as both govern bus resources) traverses the schedule by reading an element 10. Based on information read out of the element, the bus master then determines whether a threshold number 12 of flow control events has been issued by the corresponding endpoint. If the threshold number of flow control events has been reached, the bus master moves on to read the next element 18 in the schedule. If the threshold number of flow control events has not been reached, the bus master determines whether there is buffer available for a transfer by reading information 14 contained in the element. If there is no buffer available, the element is simply skipped and the bus master reads the next element. If there is buffering available, the bus master executes a transaction on the bus 16 in accordance with the type of data to be transmitted. If the device responds with a flow-control event 20, a counter 22 is decremented in the element. The consequences of a counter which reaches zero are discussed below. If the device has data to transmit, the counter in the element is not adjusted. The access pattern continues on in this fashion until certain events, discussed below, occur to stop or delay the traversal of the schedule.

Referring specifically now to the counter used by an embodiment of the present invention, it should be noted that the counter need not be contained in each of the elements. Rather, the counter could be located elsewhere in the system. In addition, although this embodiment shows a decrement of a counter when an endpoint responds with a flow control event, it is contemplated by the present invention to increment a counter until a threshold number of flow control events have been observed. Furthermore, the counter need not be linear (e.g. 2, 1, 0, reset, 2, 1, 0), but rather the counter could be circular (e.g. 2, 1, 0, 2, 1, 0) with no resetting needed. Moreover, the threshold number is programmable and can be variable or fixed, depending on the needs of the user.

As mentioned earlier, the bus master traverses the schedule executing transactions on the bus when buffering is available. In one embodiment, the present invention counts the number of flow control events issued (by decrementing a counter) such that when the counter reaches zero, the bus master "skips" that element during subsequent traversals (and so long as the counter is not reset). "Skipping" means that the bus master reads a zero in the counter for a particular element and moves on to the next element without making any attempt to execute a bus transaction to the endpoint/device associated with the element with a zero in its counter. In an embodiment with an incrementing counter rather than a decrementing counter, the element would be skipped when the bus master reads the threshold number in the counter. This algorithm is based on the idea that once an endpoint/device has issued a certain number of flow control events, it is unlikely that the endpoint will have data to move in the near future. Thus, the bus master saves bus time by

skipping elements which have issued a threshold number of flow control events and rewarding elements that are willing to move data. In an alternative embodiment, the host system and/or bus master keep track of which endpoints have issued the threshold number of flow control events and skip their respective elements altogether, not even reading the counter.

Referring now to **Figure 2** which shows a diagram of a sample schedule 24 which has four elements in a circular linked list, each element 26-32 contains information pertaining to a particular endpoint (not shown). When traversing the schedule 24, the bus master follows the pattern shown in **Figure 1**. In an embodiment, the traversal of the schedule 24 stops upon the happening of one or more events. For instance, the bus master may cease traversal at the end of a USB microframe, when an empty list is detected, or when the schedule has been disabled. Regardless of the reason for stopping, the bus master can restart traversal of the schedule 24 after sitting idle for a period of time or when the bus master is required to service the schedule in the next USB microframe. As the bus master makes its first complete traversal of the schedule 24 upon starting, the bus master will reset each of the counters 26A, 28A, 30A, and 32A to an initial value. The initial value is programmable by the user. In an embodiment wherein the counter decrements, the initial value is set to a number other than zero. After the first complete traversal, the bus master follows the pattern set forth in **Figure 1** and does not reset any of the counters until traversal ceases and then subsequently resumes.

In accordance with one embodiment of the invention, the elements 26-32 may have a field by which the bus master can designate one of the elements 26-32 as a head or beginning of the list. For instance, if the bus master begins traversal of the schedule 24 of **Figure 2** by reading element 26, the bus master can signify element 26 as the head of the list by writing an indicator into the H-bit field 26B of element 26. Once element 26 is marked as the head, the bus master proceeds to service the remainder of the list. The purpose of identifying a head is so the bus master can stop traversal of the schedule if a full traversal of the schedule 24 is made without executing any bus transactions. Thus, if the bus master arrives back at the head element 26 after a full traversal of the schedule without executing any transactions, the bus master will cease traversal for some period of time. The period of idle time may be adjustable or fixed. In addition, the idle time could increase linearly or exponentially as the number of times the bus master goes idle increases during a USB microframe. Once the bus master "wakes" from this idle state, it can resume traversal of the schedule 24 and reset all element counters 26A-32A during the first traversal of the schedule as discussed earlier.

In accordance with one aspect of the invention, the bus master may stop traversal of the schedule 24 when all of the endpoints 26-32 have issued the threshold number of flow control events. The bus master then sits idle for a period of time before restarting (and

possibly resetting counters 26A-32A) or until the end of the current USB microframe. Again, the idle time can be adjustable or fixed. In one embodiment, the idle time is fixed at 10 microseconds.

Turning now to **Figure 3,** a system is shown which is capable of implementing the method set forth above. A host 34 is shown which has a processor 36, a memory 38, and a bus master/host controller 40. The schedule 42 is located within the memory 38. It should be noted, however, that the schedule 42 could be located other than in memory 38 so long as the bus master/host controller 40 could access the schedule 42 in accordance with an embodiment of the present invention. The bus master/host controller 40 is coupled to a bus 44 with devices 46-52 attached to the bus 44. These exemplary devices 46-52 are a keyboard 46, a mouse 48, a monitor 50, and a joystick 52. Although not shown, the schedule 42 contains the list of elements, and each element contains information, including the counter, which pertains to each particular endpoint/device 46-52. Thus, the bus master/host controller 40 can access and traverse the schedule 42 in memory 38 and execute transactions on the bus 44 in accordance with information obtained from the elements in the schedule 42. Moreover, the bus master/host controller 40 can decrement the counter of any element in memory 38 which represents a device that issues a flow control event. When the threshold number of flow control events has been issued by a particular endpoint/device, the bus master/host controller 40 suspends service to that endpoint/device by "skipping" the device as discussed above.

As mentioned earlier, the counter can count in either a linear fashion or a circular fashion. Moreover, the schedule list of elements need not be a circular linked list. Rather, the schedule could be an array of elements through which the bus master traverses either by row, column, or any other suitable method.

In addition, although the preferred embodiment described herein is directed to USB, it will be appreciated by those skilled in the art that the teaching contained herein can be applied to other systems.

It is to be understood that even though numerous characteristics and advantages of various embodiments of the present invention have been set forth in the foregoing description, together with details of the structure and function of various embodiments of the invention, this disclosure is illustrative only. Changes may be made in detail, especially matters of structure and management of parts, without departing from the scope of the present invention as expressed by the broad general meaning of the terms of the appended claims.